Amendments to the Specification

1. Please replace paragraph [0002] with the following amended paragraph [0002]:

[0002] With the explosion in Internet 40, as shown in FIG. 1, access and usage individuals have discovered and become dependent upon the availability of large amounts of information as well and the ability to buy and sell goods and services. As shown in FIG. 1, a typical Internet user would have a browser installed in his personal computer (PC) 10 or server 20 such as Internet Explorer™ or Netscape™. Using this browser, the user would access an Internet service provider, such as America-On-Line (AOL™) (not shown), via a modem over the local public switched telephone network (PSTN), a cable network or satellite link. Once logged onto an Internet web server (web server) 30, the user may utilize one of the many search engines, such as Yahoo™ or Lycos™, to specify search terms. The user could also log onto a web server 30 and view the products or services available for sale or receive the information desired.

2. Please replace paragraph [0003] with the following amended paragraph [0003]:

[0003] FIG. 2 illustrates the software and hardware involved for communications between a server 20 and a web server 30. Server 20 would contain application software 200, such as, but not limited to, a browser, communicating to a network protocol 210, such as, but not limited to, TCP/IP (Transmission Control Protocol/ Internet Protocol) or UDP (User Datagram Protocol), which in turn would communicate to a network interface 220. The network interface 220 may be, but is not limited to, any type of serial or parallel modem. The network interface 220 would communicate to the network/Internet 40 which in turn would interface to web server 30. Again, within web server 30, a

network interface 230, such as a serial or parallel modem, would communicate to the network protocol 240, such as, but not limited to, TCP/IP or UDP. Thereafter, communications would be established with an application 250 which may be a search engine or any other type of web application.

3. Please replace paragraph [0006] with the following amended paragraph [0006]:

[0006] Still referring to FIG. 3, once a "hacker" has embedded the zombie 300 in the servers 20 all that is needed to initiate the DDoS attack is a denial of service initiator 310. This denial of service initiator 310 may be a message from the "hacker" or a specific time of day. It should be noted that FIG. 3 is identical to FIG. 2 with the exception of the zombie 300 and the denial of service initiator 310. This allows a large and important web server to be easily disabled from use, costing, in many cases, millions of dollars in lost revenue.

4. Please replace paragraph [0007] with the following amended paragraph [0007]:

[0007] Therefore, a system, method, and computer program is needed that ~~well~~ will detect the presence of zombie applications and block them from launching a massive number of packets for delivery to a web server. This system, method, and computer program must detect and block the zombie packets before they can cause any denial of service to a web server. Further, this system, method, and computer program must be compatible with existing communications protocols involved in packet switched networks. Further, the system, method, and computer program must be easy to install and not interfere with normal packet transmission and reception.

5. Please replace paragraph [0024] with the following amended paragraph [0024]:

[0024] FIG. 5 is a modular configuration diagram of the ZADAR intermediate driver 400 utilized in an example embodiment of the present invention and further detailed in the flowcharts illustrated in FIGs. 6 through 9. The ZADAR intermediate driver 400 comprises three major components. The first component is a transmit (Tx) algorithm 500 used to monitor incoming packets. The Tx algorithm 500 is discussed in further detail in reference to FIGs. 6 and 9. The second major component is the receive (Rx) algorithm 520, which is discussed in further detail in reference to Figs. 7 and 9. The third major components in the ZADAR intermediate driver 400 is the monitor code 510, which is discussed in further detail in FIG. 9 8. As shown in FIG. 5 the Tx algorithm 500 receives packets from the network protocol 210 and transmits them to the network interface 220. Further, the Rx algorithm 520 receives packets from the network interface 220 and transmits them to the network protocol 210. Both the Tx algorithm 500 and Rx algorithm 520 communicate to the monitor code 510. The monitor code 510 does not actively send or receive packets of information, but does monitor the activities of applications 200 and zombies 300 through information received from the Tx algorithm 500 and the Rx algorithm 520.

6. Please replace paragraph [0028] with the following amended paragraph [0028]:

[0028] FIG. 7 is a flowchart illustrating the logic involved in the receive (Rx) algorithm 520, shown in FIG. 5, in an example embodiment of present invention. The Rx algorithm 520 begins execution in operation 700 and immediately proceeds to operation 710. In operation 710, a packet is received by the Rx algorithm ~~526~~ 520 from the network

interface 220, either from an application 200 or a zombie 300. Thereafter, in operation 720, it is determined if the packet is from a known application. Operation 720 is further detailed in the discussion provided in reference to FIG. 9. If the packet is determined in operation 720 to be from a known application, then processing proceeds to operation 730. In operation 730, the application is registered and processing proceeds to operation 740 where the usage of the network is tracked by storing the destination, packet size and packet count using the monitor code 510.

7. Please replace paragraph [0030] with the following amended paragraph [0030]:

[0030] However, still referring to FIG. 7, if in operation 760 it is determined the packet is not from a known zombie 300, then processing proceeds to operation 740. In operation 740, as previously discussed, the network usage is stored based upon the destination address, packet size, and packet count using the monitor code 510. Thereafter, processing proceeds to operation 750 where the packet is passed to the network protocol 210 for transmission to the desired application 200. Thereafter, processing proceeds to operation 780 where the Rx algorithm 520 terminates execution.

8. Please replace paragraph [0037] with the following amended paragraph [0037]:

[0037] FIG. 9 is a flowchart further detailing the logic involved in operations 620 and 650 illustrated in FIG. 6 and 720 and 750 illustrated in FIG. 7, in an example embodiment of the present invention. The logic involved in FIG. 9 attempts to determine whether a particular application is a known good application 200 or a zombie 300 based upon the destination port specified. Execution begins in operation ~~906~~ 900 and

immediately proceeds to operation 905 where the destination port number provided by the TCP/IP or UDP header is checked. Thereafter, in operation 910 it is determined whether the particular destination port is from a known good port. If the port number is known to be a good port then processing proceeds to operation 945 where the classification process is completed and processing terminates in operation 950.

9. Please replace paragraph [0038] with the following amended paragraph [0038]:

[0038] Still referring to FIG. 9, if it is determined that the port number from the TCP/IP or UDP header is not a known good port then processing proceeds to operation 915. In operation 915 ~~515~~, it is determined whether the port number in question is a known ~~the~~ zombie port. If the port number is known to be from a zombie port then processing proceeds to operation 940 where packets received counter is incremented for the connection value and processing proceeds, as previously discussed, to operation 945. However, if in operation 915 the port number is not a known zombie port then processing proceeds to operation 920. In operation 920 it is determined whether the source port number from the TCP/IP or UDP header is a known zombie port. If the source port number is from a known zombie port then again processing proceeds to operation 940. However, if the source port number is not known to be a zombie port then processing proceeds to operation 925. In operation 925 the IP address, IP destination address, destination port number, and source port number are hashed to form a single connection value. This single connection value will serve as a unique identifier for this particular application 200. Thereafter, processing proceeds to operation 930 where the connection value computed in operation 925 is checked against a list to determine if it is present. If

the connection value is not present in the list then processing proceeds to operation 935 where it is added to the list and again processing then proceeds to operation 940.